UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/632,062	07/31/2003	Michel Betancourt	RSW920030145US1	2039

36736 7590 10/04/2006

DUKE W. YEE
YEE & ASSOCIATES, P.C.
P.O. BOX 802333
DALLAS, TX 75380

EXAMINER	
PUENTE, EMERSON C	
ART UNIT	**PAPER NUMBER**
2113	

DATE MAILED: 10/04/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/632,062	BETANCOURT ET AL.	
	Examiner	Art Unit	
	Emerson C. Puente	2113	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 31 July 2003.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-22 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-22 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on 31 July 2003 is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date 7/31/03.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

This action is made **Non-Final.**

Claims 1-22 have been examined.

Specification

The disclosure is objected to because of the following informalities:

The use of trademarks, such as JAVA, has been noted in this application. It should be capitalized wherever it appears and be accompanied by the generic terminology.

Although the use of trademarks is permissible in patent applications, the proprietary nature of the marks should be respected and every effort made to prevent their use in any manner which might adversely affect their validity as trademarks.

Examiner suggest capitalizing each letter of the word or include a proper trademark symbol, such as ™ or © following the word.

Appropriate correction is required.

Claim Rejections - 35 USC § 101

35 U.S.C. 101 reads as follows:

Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claims 20-22 are rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter.

In regards to claim 20-22, the claimed "computer readable medium" as described in the specification page(s) 11, lines 3-5, includes, among other examples, transmission-type media, such as digital and analog communication links, wired or wireless communications links using transmission forms, such as radio frequency and light wave transmission, which is nonstatutory. As such, the claim is not limited to statutory subject matter and is therefore non-statutory.

Claim Rejections - 35 USC § 103

The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

Claims 1-8 and 10-22 are rejected under 35 U.S.C. **103(a)** as being unpatentable over US Patent No. 5,590,335 of Dubourreau et al. referred hereinafter "Dubourreau" in view of Applicant's Admitted Prior Art referred hereinafter "AAPA".

In regards to claim 1, Dubourreau discloses system for analyzing thread deadlocks, comprising:

a thread analyzer tool, wherein the tool analyzes a thread dump to automatically identify thread deadlocks. Dubourreau discloses when a deadlock occurs, one can call on a tool for analyzing memory dumps (see column 6 lines 63-67).

wherein the tool identifies threads that are in a self wait condition and threads that are in a circular wait condition. Dubourreau disclose a traditional process that initializes a list, and follows a list of steps that adds threads to the list. When a thread appears twice on the list, a cycle is identified, indicating a deadlock condition (see column 4 lines 35-46). If the cycle displays only one thread, a self wait condition is identified and if a cycle includes more than one thread (such as the one described in column 1 lines 34-41), then a circular wait condition is identified.

However, Dubourreau fails to explicitly disclose:

analyzing deadlocks in a Java virtual machine.

However, AAPA disclose Java virtual machine supports multithreading, which is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Dubourreau onto a Java virtual machine, as disclosed in AAPA. A person of ordinary skill in the art would have been motivated to combine the teachings because Java virtual machine, as per teachings of AAPA, supports multithreading, and hence is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16), and analyzing deadlocks, as per teachings of Dubourreau, enables identification of the cause of the deadlock, enabling modification of programming more effectively (see column 3 lines 45-52).

In regards to claim 2, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses a user interface that allows a user to specify criteria, wherein the tool excludes threads that do not meet the criteria from identification as deadlocked threads. Dubourreau discloses a user can interactively, indicating a user interface, call on a tool when a deadlock occurs to analyzes a memory dump (see column 6 lines 63-67). The tool runs a process that specifies criteria that identifies threads in a cycle (see column 4 lines 30), indicating deadlock threads. Threads not in a cycle are threads not in a deadlock condition, indicating threads that do not meet the criteria and hence excluded from identification as deadlock threads.

In regards to claim 3, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the tool analyzes a thread dump file in offline mode. Dubourreau discloses analyzing memory dumps when there is a deadlock and the machine becomes unavailable (see column 6 lines 63 to column 7 line 3).

In regards to claim 4, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the tool obtains a thread dump from a running information processing system. Dubourreau discloses analyzing memory dumps of a machine that is in operation (see column 6 lines 59-63).

In regards to claim 5, Dubourreau discloses a method of analyzing thread deadlocks, comprising the steps of:

obtaining a thread dump. Dubourreau discloses when a deadlock occurs, one can call on a tool for analyzing memory dumps (see column 6 lines 63-67).

analyzing the thread dump to automatically identify threads in a deadlock condition, wherein threads in a circular wait condition and threads in a self wait condition are identified. Dubourreau disclose a traditional process that initializes a list, and follows a list of steps that adds threads to the list. When a thread appears twice on the list, a cycle is identified, indicating a deadlock condition (see column 4 lines 35-46). If the cycle displays only one thread, a self wait condition is identified and if a cycle includes more than one thread (such as the one described in column 1 lines 34-41), then a circular wait condition is identified.

However, Dubourreau fails to explicitly disclose:

analyzing deadlocks in a Java virtual machine.

However, AAPA disclose Java virtual machine supports multithreading, which is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Dubourreau onto a Java virtual machine, as disclosed in AAPA. A person of ordinary skill in the art would have been motivated to combine the teachings because Java virtual machine, as per teachings of AAPA, supports multithreading, and hence is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16), and analyzing deadlocks, as per teachings of Dubourreau, enables identification of the cause of the deadlock, enabling modification of programming more effectively (see column 3 lines 45-52).

In regards to claim 6, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses a user interface that allows a user to specify criteria, wherein threads that do not meet the criteria are filtered. Dubourreau discloses a user can interactively, indicating a user interface, call on a tool when a deadlock occurs to analyze a memory dump (see column 6 lines 63-67). The tool runs a process that specifies criteria that identifies which threads form a cycle (see column 4 lines 30), indicating deadlock threads. Threads not in a cycle are threads not in a deadlock condition, indicating threads that do not meet the criteria that are filtered.

In regards to claim 7, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein filtered threads are excluded from identification as threads in a self wait condition. The tool runs a process that specifies criteria that identifies which threads form a cycle (see column 4 lines 30), indicating deadlock threads. Threads not in a cycle are threads not in a deadlock condition, indicating threads excluded from identification as threads in a self wait condition.

In regards to claim 8, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the tool analyzes a thread dump file in offline mode. Dubourreau discloses analyzing memory dumps when there is a deadlock and the machine becomes unavailable (see column 6 lines 63 to column 7 line 3).

In regards to claim 10, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the step of analyzing the thread dump to automatically identify threads in a deadlock condition includes the steps of:

(a) identifying threads that own resources. Dubourreau discloses searching for threads that holds a lock (see column 4 line 25-27).

(b) identifying threads that are waiting on resources. Dubourreau discloses determine if the thread is waiting for another lock (see column 4 lines 25-27).

comparing the results from steps (a) and (b) to identify threads in a circular wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29).

In regards to claim 11, Dubourreau discloses a method of analyzing thread deadlocks, comprising the steps of:

obtaining a thread dump file. Dubourreau discloses when a deadlock occurs, one can call on a tool for analyzing memory dumps (see column 6 lines 63-67).

identifying waiting threads. Dubourreau discloses determine if the thread is waiting for another lock (see column 4 lines 25-27).

identifying locking threads. Dubourreau discloses searching for threads that holds a lock (see column 4 line 25-27).

comparing waiting threads and locking threads to identify threads in a self wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29). If the cycle displays only one thread, a self wait condition is identified.

However, Dubourreau fails to explicitly disclose:

analyzing deadlocks in a Java virtual machine.

However, AAPA disclose Java virtual machine supports multithreading, which is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Dubourreau onto a Java virtual machine, as disclosed in AAPA. A person of ordinary skill in the art would have been motivated to combine the teachings because Java virtual machine, as per teachings of AAPA, supports multithreading, and hence is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16), and analyzing deadlocks, as per teachings of Dubourreau, enables identification of the cause of the deadlock, enabling modification of programming more effectively (see column 3 lines 45-52).

In regards to claim 12, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses comparing waiting threads and locking threads to identify threads in a circular wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29). If the cycle includes more than one thread (such as the one described in column 1 lines 34-41), then a circular wait condition is identified.

In regards to claim 13, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the step of obtaining a thread dump file comprises obtaining a thread dump from a live JVM. Dubourreau discloses analyzing memory dumps of a machine that is in operation (see column 6 lines 59-63) and AAPA discloses a JVM (see page 3 lines 5-7 and page 4 lines 15-16).

In regards to claim 14, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the step of obtaining a thread dump file comprises opening an existing thread dump file. Dubourreau discloses when a deadlock occurs, one can call on a tool for analyzing memory dumps (see column 6 lines 63-67).

In regards to claim 15, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein the existing thread dump file is analyzed in offline mode. Dubourreau discloses analyzing memory dumps when there is a deadlock and the machine becomes unavailable (see column 6 lines 63 to column 7 line 3).

In regards to claim 16, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein a user interface allows a user to choose rules, and wherein the rules are used to exclude threads from being identified as in a deadlock condition. Dubourreau discloses a user can interactively, indicating a user interface, call on a tool when a deadlock occurs to analyzes a memory dump (see column 6 lines 63-67). The tool runs the process with rules that identifies threads in a cycle (see column 4 lines 30), indicating deadlock threads. Threads not in a cycle are threads not in a deadlock condition, indicating threads that do not follows rules and hence excluded from identification as deadlock threads.

In regards to claim 17, Dubourreau discloses a system for analyzing thread deadlocks, comprising the steps of:

means for obtaining a thread dump file. Dubourreau discloses when a deadlock occurs, one can call on a tool for analyzing memory dumps (see column 6 lines 63-67).

means for identifying waiting threads. Dubourreau discloses determine if the thread is waiting for another lock (see column 4 lines 25-27).

means for identifying locking threads. Dubourreau discloses searching for threads that holds a lock (see column 4 line 25-27).

means for comparing waiting threads and locking threads to identify threads in a self wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29). If the cycle displays only one thread, a self wait condition is identified.

However, Dubourreau fails to explicitly disclose:

analyzing deadlocks in a Java virtual machine.

However, AAPA disclose Java virtual machine supports multithreading, which is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Dubourreau onto a Java virtual machine, as disclosed in AAPA. A person of ordinary skill in the art would have been motivated to combine the teachings because Java virtual machine, as per teachings of AAPA, supports multithreading, and hence is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16), and analyzing deadlocks, as per teachings of Dubourreau, enables identification of the cause of the deadlock, enabling modification of programming more effectively (see column 3 lines 45-52).

In regards to claim 18, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses means for comparing waiting threads and locking threads to identify threads in a circular wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29). If the cycle includes more than one thread (such as the one described in column 1 lines 34-41), then a circular wait condition is identified.

In regards to claim 19, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein thread deadlocks are analyzed in offline mode. Dubourreau discloses analyzing memory dumps when there is a deadlock and the machine becomes unavailable (see column 6 lines 63 to column 7 line 3).

In regards to claim 20, Dubourreau discloses a computer program product in a computer readable medium for analyzing thread deadlocks in a Java virtual machine, comprising:

first instructions for obtaining a thread dump file. Dubourreau discloses when a deadlock occurs, one can call on a tool for analyzing memory dumps (see column 6 lines 63-67).

second instructions for identifying waiting threads. Dubourreau discloses determine if the thread is waiting for another lock (see column 4 lines 25-27).

third instructions for identifying locking threads. Dubourreau discloses searching for threads that holds a lock (see column 4 line 25-27).

fourth instructions for comparing waiting threads and locking threads to identify threads in a self wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29). If the cycle displays only one thread, a self wait condition is identified.

However, Dubourreau fails to explicitly disclose:

analyzing deadlocks in a Java virtual machine.

However, AAPA disclose Java virtual machine supports multithreading, which is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to incorporate the teachings of Dubourreau onto a Java virtual machine, as disclosed in AAPA. A person of ordinary skill in the art would have been motivated to combine the teachings because Java virtual machine, as per teachings of AAPA, supports multithreading, and hence is vulnerable to deadlock situations (see page 3 lines 5-7 and page 4 lines 15-16), and analyzing deadlocks, as per teachings of Dubourreau, enables identification of the cause of the deadlock, enabling modification of programming more effectively (see column 3 lines 45-52).

In regards to claim 21, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses fifth instructions for comparing waiting threads and locking threads to identify threads in a circular wait condition. Dubourreau discloses detecting a cycle in the graph, indicating that a deadlock has been found involving the processes (threads) and resources in said cycle (see column 4 lines 27-29). If the cycle includes more than one thread (such as the one described in column 1 lines 34-41), then a circular wait condition is identified.

In regards to claim 22, Dubourreau in view of AAPA discloses the claim limitations as discussed above. Dubourreau further discloses wherein thread deadlocks are analyzed in offline mode. Dubourreau discloses analyzing memory dumps when there is a deadlock and the machine becomes unavailable (see column 6 lines 63 to column 7 line 3).

Claim 9 is rejected under 35 U.S.C. **103(a)** as being unpatentable over Dubourreau in view of AAPA, and in further view of US Patent Application No. 2003/0023656 of Hutchison.

In regards to claim 9, Dubourreau in view of AAPA discloses the claim limitations as discussed above. However, Dubourreau fails to disclose wherein a matrix is populated with threads owning resources and threads waiting on resources, and wherein the matrix is used to identify threads in a circular wait condition.

Hutchison discloses a matrix is populated with threads owning resources and threads waiting on resources. Hutchison disclose analyzing deadlocks mathematically via a node-arc matrix with rows indicating From (F), indicating threads waiting on resources, and columns indicating To (T), indicating thread owning resources (see figure 5b and 6b and paragraph 67).

wherein the matrix is used to identify threads in a circular wait condition. Hutchison discloses any cyclic path is indicated by a non-zero number in the primary diagonal (see paragraph 68). Thus, a circular wait condition is identified when there is a non zero number in the primary diagonal of a matrix for two or more step paths (see figure 5b and paragraph 69).

It would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Dubourreau, AAPA, and Hutchison to include a matrix populated with threads owning resources and threads waiting on resources, and wherein the matrix is used to identify threads in a circular wait condition. A person of ordinary skill in the art would have been motivated to combine the teachings because Dubourreau is concerned with identifying cycles that resulted in a deadlock (see column 4 lines 27-29) and matrices, as per teaching of Hutchison, is well known in deadlock analysis (see paragraph 71) to identify cyclic paths (see paragraph 68).

Conclusion

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

See Form PTO-892.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Emerson C. Puente whose telephone number is (571) 272-3652. The examiner can normally be reached on 8-5 M-F.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Robert W. Beausoliel can be reached on (571) 272-3645. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

Information regardsing the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Emerson Puente

Emerson Puente
Examiner
AU 2113